



Taller de Programación



Agenda



Evolución de Arquitecturas

Conceptos de Concurrencia

Ejemplos

Ambiente CMRE



Situaciones Cotidianas



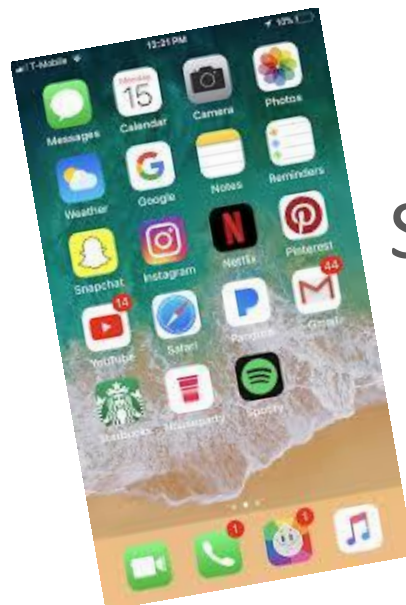
Navegadores



Sistemas Operativos



Cuentas Bancarias



Smartphones



Qué características comunes hay en estos ejemplos?



Evolución de las Arquitecturas

1 núcleo de
procesamiento



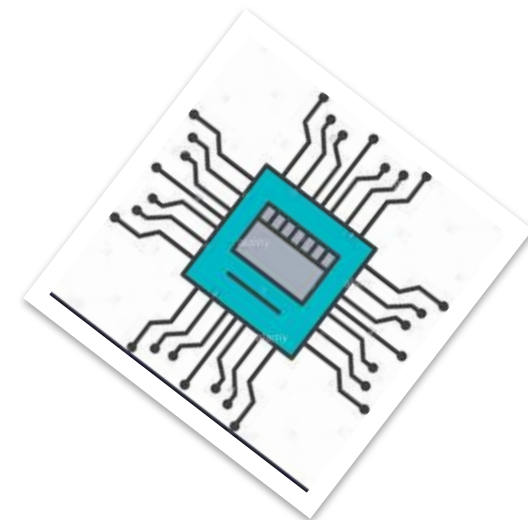
1980

2,4,8 núcleos de
procesamiento



2000

2,4 millones de
núcleos de
procesamiento
primera en el Top
500



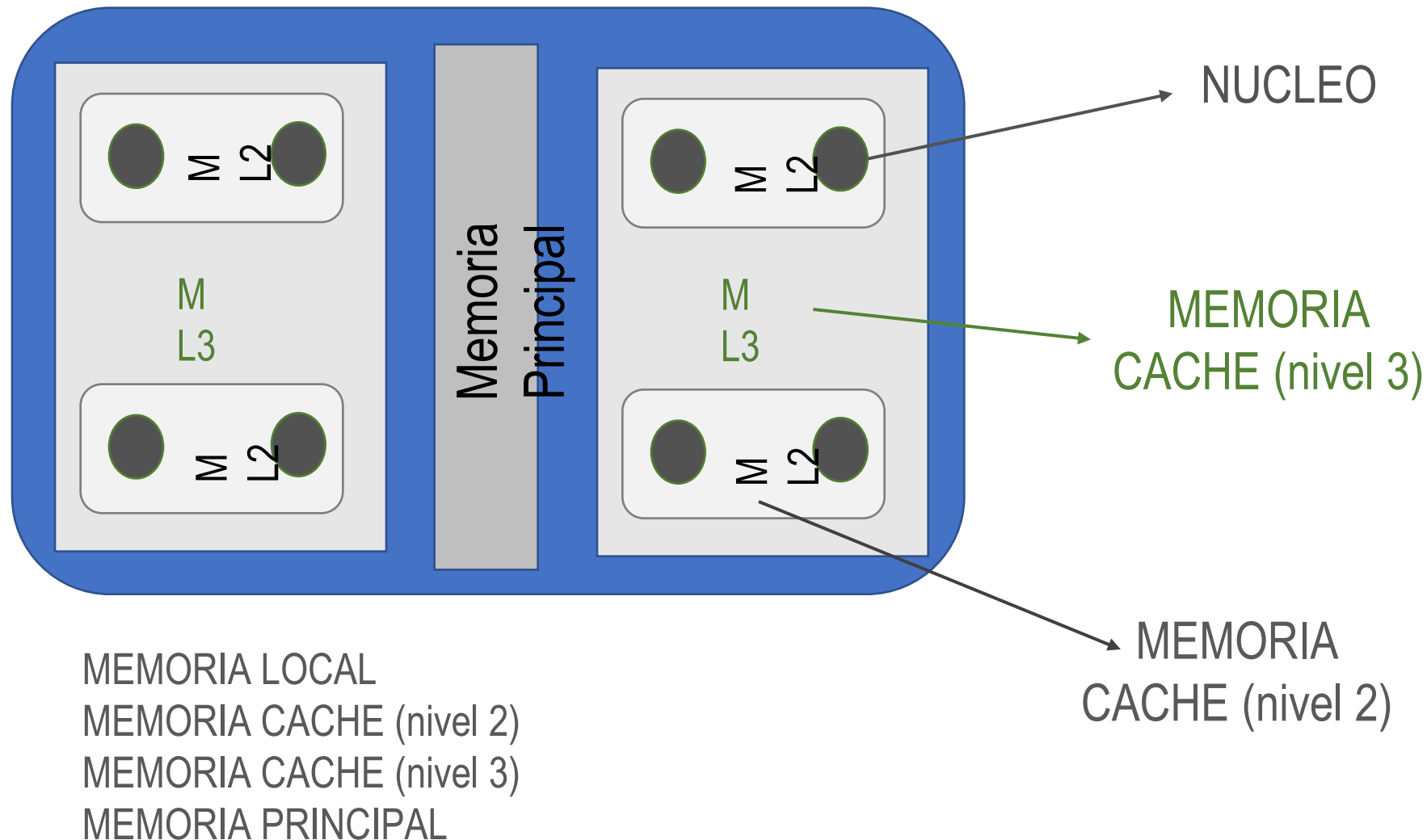
2019



Qué es un procesador
de 8 núcleos?

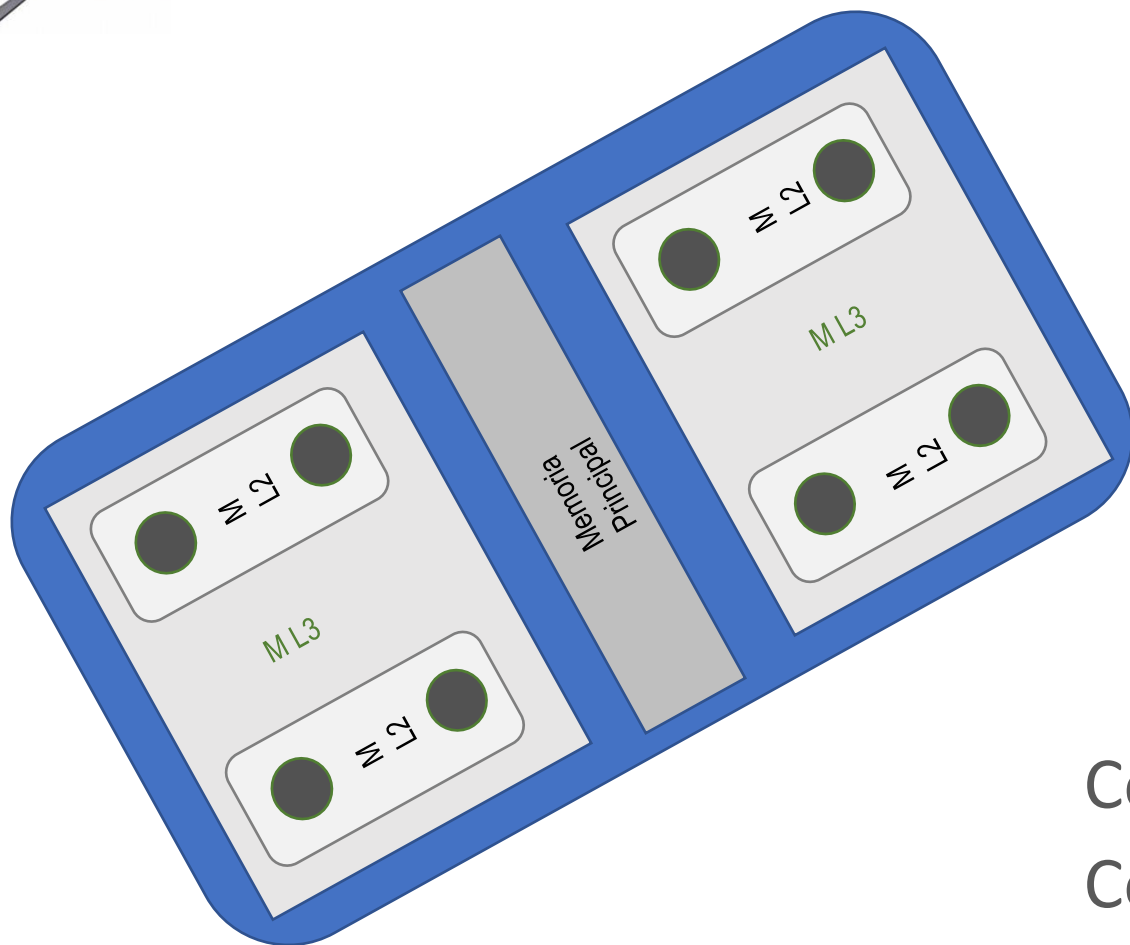


Evolución de las Arquitecturas

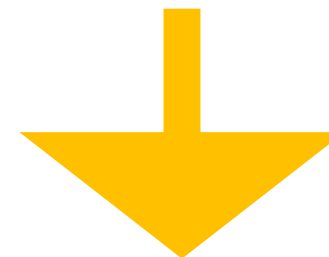




Evolución de las Arquitecturas



Para poder **explotar** este hardware los programas deben ser concurrentes



Cómo se comunican los procesos?

Cómo coordinan?

Cómo se escribe correctamente el programa?



Concepto de Concurrency



Un **programa concurrente** se divide en tareas (2 o mas), las cuales se ejecutan al mismo tiempo y realizan acciones para cumplir un objetivo común. Para esto pueden: compartir recursos, coordinarse y cooperar

Características: **comunicación - sincronización**



Se ha vuelto un concepto clave en las Ciencias de la Computación.

Influye tanto al hardware como al software.



Ejemplo



Automóviles: son los procesos que se ejecutan



Carriles: son los múltiples procesadores

Los automóviles deben sincronizarse
para no chocar

Objetivo: examinar los tipos de autos (procesos), trayecto a recorrer (programas), caminos (hardware), y reglas (comunicación y sincronización).



Ejemplo



Suponga que una pareja comparte su cuenta bancaria. En un momento los dos integrantes de la pareja van a un cajero y extraen 1000 pesos. Es correcto el siguiente código?

Variable compartida saldo



Cómo se protege saldo?

```
Integrante 1:  
{  
    accede a la cuenta  
    saldo:= saldo - 1000;  
}
```

```
Integrante 2:  
{  
    accede a la cuenta  
    saldo:= saldo - 1000;  
}
```



Ejemplo



Suponga que una pareja comparte su cuenta bancaria. En un momento los dos integrantes de la pareja van a un cajero y extraen 1000 pesos.



Cómo se
protege saldo?

Cualquier lenguaje que
brinde concurrencia
debe proveer
mecanismos para
comunicar y **sincronizar**
procesos.



En este caso quiero **proteger**
el acceso a la variable
compartida (dos procesos
no accedan al mismo
tiempo, sincronicen)



Semáforos P y V
Monitores
Pasaje de Mensajes



Ejemplo



Suponga que una pareja comparte su cuenta bancaria. En un momento los dos integrantes de la pareja van a un cajero y extraen 1000 pesos. Es correcto el siguiente código?

Variable compartida saldo



Se puede mejorar?

```
Integrante 1:  
{ P(saldo)  
  accede a la cuenta  
  saldo:= saldo - 1000;  
  V(saldo)  
}
```

```
Integrante 2:  
{ P(saldo)  
  accede a la cuenta  
  saldo:= saldo - 1000;  
  V(saldo)  
}
```



Ejemplo



Suponga que una pareja comparte su cuenta bancaria. En un momento los dos integrantes de la pareja van a un cajero y extraen 1000 pesos. Es correcto el siguiente código?

Variable compartida saldo

Integrante 1:

```
{ accede a la cuenta
  P(saldo)
  saldo:= saldo - 1000;
  V(saldo)
}
```

Integrante 2:

```
{ accede a la cuenta
  P(saldo)
  saldo:= saldo - 1000;
  V(saldo)
}
```



Ejemplo



En un programa existen 3 procesos, un arreglo de longitud M y un valor N y se quiere calcular cuantas veces aparece el valor N en el arreglo.



cont

V



Proceso 1

Proceso 2

Proceso 3

Variable compartida cont y V



Se puede mejorar?

```
Proceso 1:
{inf:=...; sup:= ...;

  for i:= inf to sup do
    if v[i] = N then
      cont:= cont + 1;
  }
```

```
Proceso 2:
{inf:=...; sup:= ...;

  for i:= inf to sup do
    if v[i] = N then
      cont:= cont + 1;
  }
```

```
Proceso 3:
{inf:=...; sup:= ...;

  for i:= inf to sup do
    if v[i] = N then
      cont:= cont + 1;
  }
```



Ejemplo



En un programa existen 3 procesos, un arreglo de longitud M y un valor N y se quiere calcular cuantas veces aparece el valor N en el arreglo.



cont

V



Proceso 1

Proceso 2

Proceso 3

Variable compartida cont y V



Se puede mejorar?

```
Proceso 1:
{inf:=...; sup:= ...;
  P(cont)
  for i:= inf to sup do
    if v[i] = N then
      cont:= cont + 1;
  V(cont)
}
```

```
Proceso 2:
{inf:=...; sup:= ...;
  P(cont)
  for i:= inf to sup do
    if v[i] = N then
      cont:= cont + 1;
  V(cont)
}
```

```
Proceso 3:
{inf:=...; sup:= ...;
  P(cont)
  for i:= inf to sup do
    if v[i] = N then
      cont:= cont + 1;
  V(cont)
}
```



Ejemplo



En un programa existen 3 procesos, un arreglo de longitud M y un valor N y se quiere calcular cuantas veces aparece el valor N en el arreglo.



cont

V



Proceso 1

Proceso 2

Proceso 3

Variable compartida cont y V



Se puede mejorar?

```
Proceso 1:
{inf:=...; sup:= ...;
  for i:= inf to sup do
    if v[i] = N then
      P(cont)
      cont:= cont + 1;
      V(cont)
}
```

```
Proceso 2:
{inf:=...; sup:= ...;
  for i:= inf to sup do
    if v[i] = N then
      P(cont)
      cont:= cont + 1;
      V(cont)
}
```

```
Proceso 3:
{inf:=...; sup:= ...;
  for i:= inf to sup do
    if v[i] = N then
      P(cont)
      cont:= cont + 1;
      V(cont)
}
```



Ejemplo



En un programa existen 3 procesos, un arreglo de longitud M y un valor N y se quiere calcular cuantas veces aparece el valor N en el arreglo.



cant

V



Proceso 1

Proceso 2

Proceso 3

Variable compartida **cant** y **V**

Proceso 1:

```
{inf:=...; sup:= ...; cant:=0
  for i:= inf to sup do
    if v[i] = N then
      cant:= cant + 1;
  P(cant)
  cont:= cont + cant;
  V(cant)
}
```

Proceso 2:

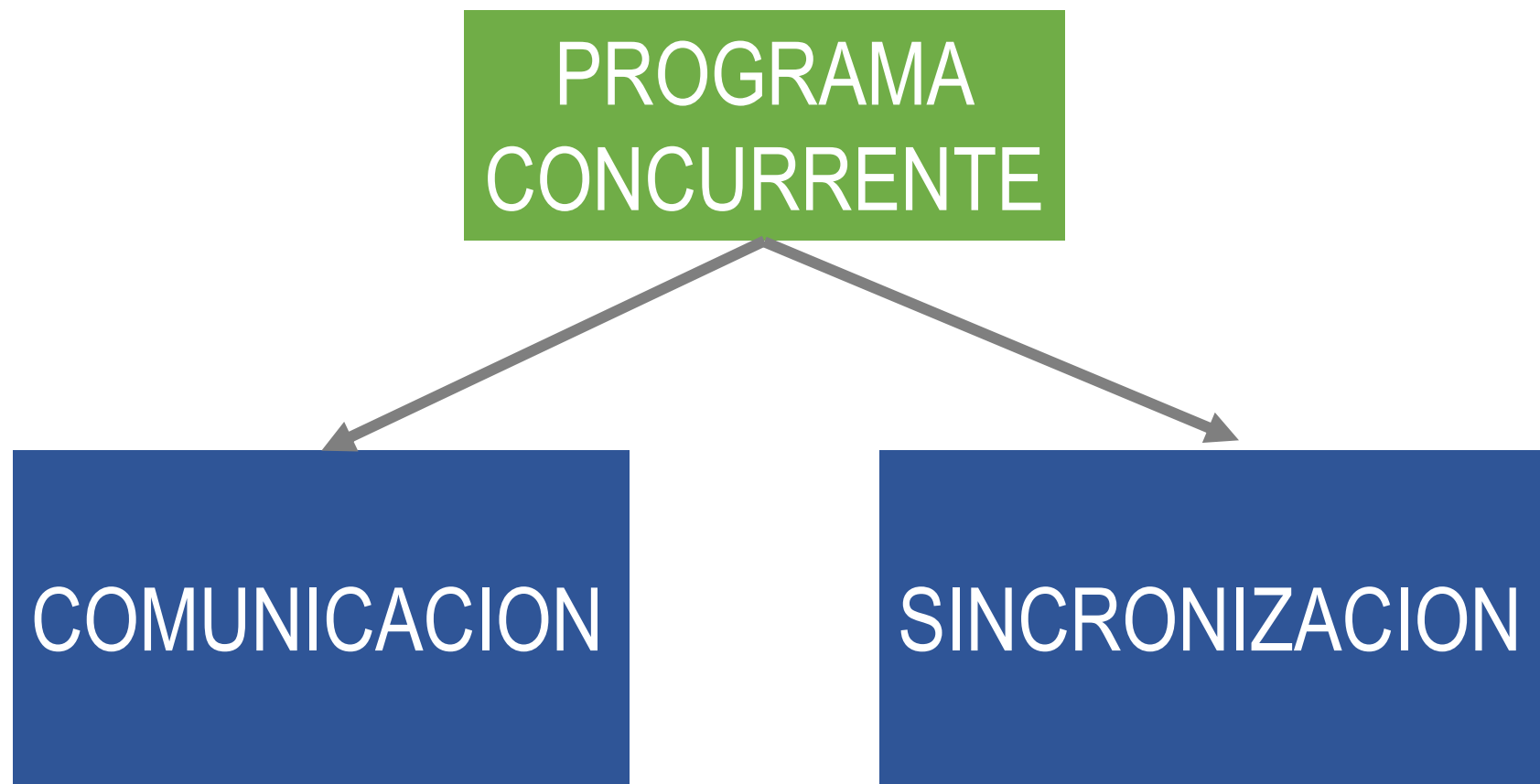
```
{inf:=...; sup:= ...; cant:=0
  for i:= inf to sup do
    if v[i] = N then
      cant:= cant + 1;
  P(cant)
  cont:= cont + cant;
  V(cant)
}
```

Proceso 3:

```
{inf:=...; sup:= ...; cant:=0
  for i:= inf to sup do
    if v[i] = N then
      cant:= cant + 1;
  P(cant)
  cont:= cont + cant;
  V(cant)
}
```

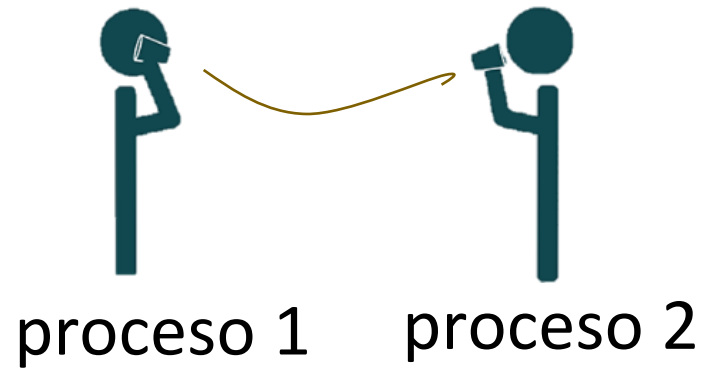



Programa Concurrente





Programa Concurrente - Comunicación

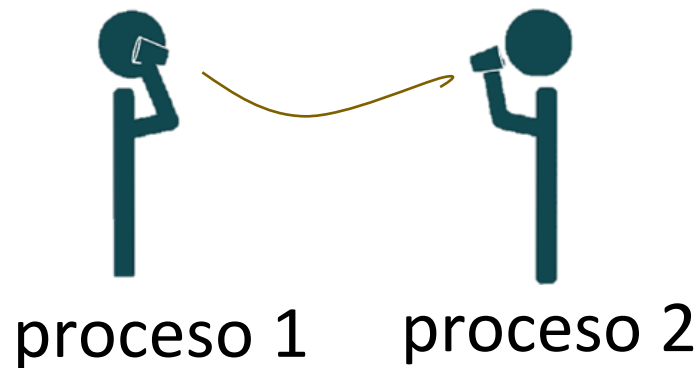


ENVIO DE
MENSAJES

MEMORIA
COMPARTIDA



Programa Concurrente - Comunicación



ENVIO DE
MENSAJES

MEMORIA
COMPARTIDA

un

de

Forma

mensaje

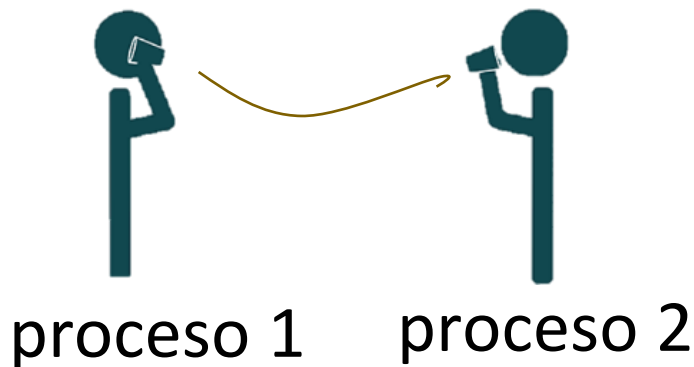
Origen
Destino
Contenido

- Es necesario establecer un canal (lógico o físico) para transmitir información entre procesos.
- También el lenguaje debe proveer un protocolo adecuado.
- Para que la comunicación sea efectiva los procesos deben “saber” cuándo tienen mensajes para leer y cuando deben transmitir mensajes.

**ENVIAR
RECIBIR**



Programa Concurrente - Comunicación



ENVIO DE
MENSAJES

MEMORIA
COMPARTIDA

Recurso
Compartido



LIBRE?

si

Bloqueo
Uso
Libero

- Los procesos intercambian información sobre la memoria compartida o actúan coordinadamente sobre datos residentes en ella.
- Lógicamente no pueden operar simultáneamente sobre la memoria compartida, lo que obliga a bloquear y liberar el acceso a la memoria.
- La solución más elemental es una variable de control que habilite o no el acceso de un proceso a la memoria compartida.

**BLOQUEAR
DESBLOQUEA**

R



Programa Concorrente

Programa Paralelo

```
(meta name="description" content="HTML tutorial")
(meta name="author" content="Andrew")
(meta name="copyright" content="2000-2011 and beyond...")
(meta name="robots" content="all")
(meta name="viewport" content="width=780",
base target="_parent")
```

```
<script>  
var target = "top";  
<style type="text/css" media="all">  
(link rel="stylesheet" type="text/css" href="/print.css" media="print")</style>  
(link rel="shortcut icon" type="image/x-icon" href="/favicon.ico")</link>  
(link rel="search" type="application/xhtml+xml" href="source-search.xml")</script>
```

```
<meta name="description" content="And...>
<meta name="author" content="And...>
<meta name="copyright" content="And...>
<meta name="robots" content="all">
<meta name="viewport" content="width=device-width, height=device-height, initial-scale=1.0, maximum-scale=1.0">
<base href="/" />
```



```

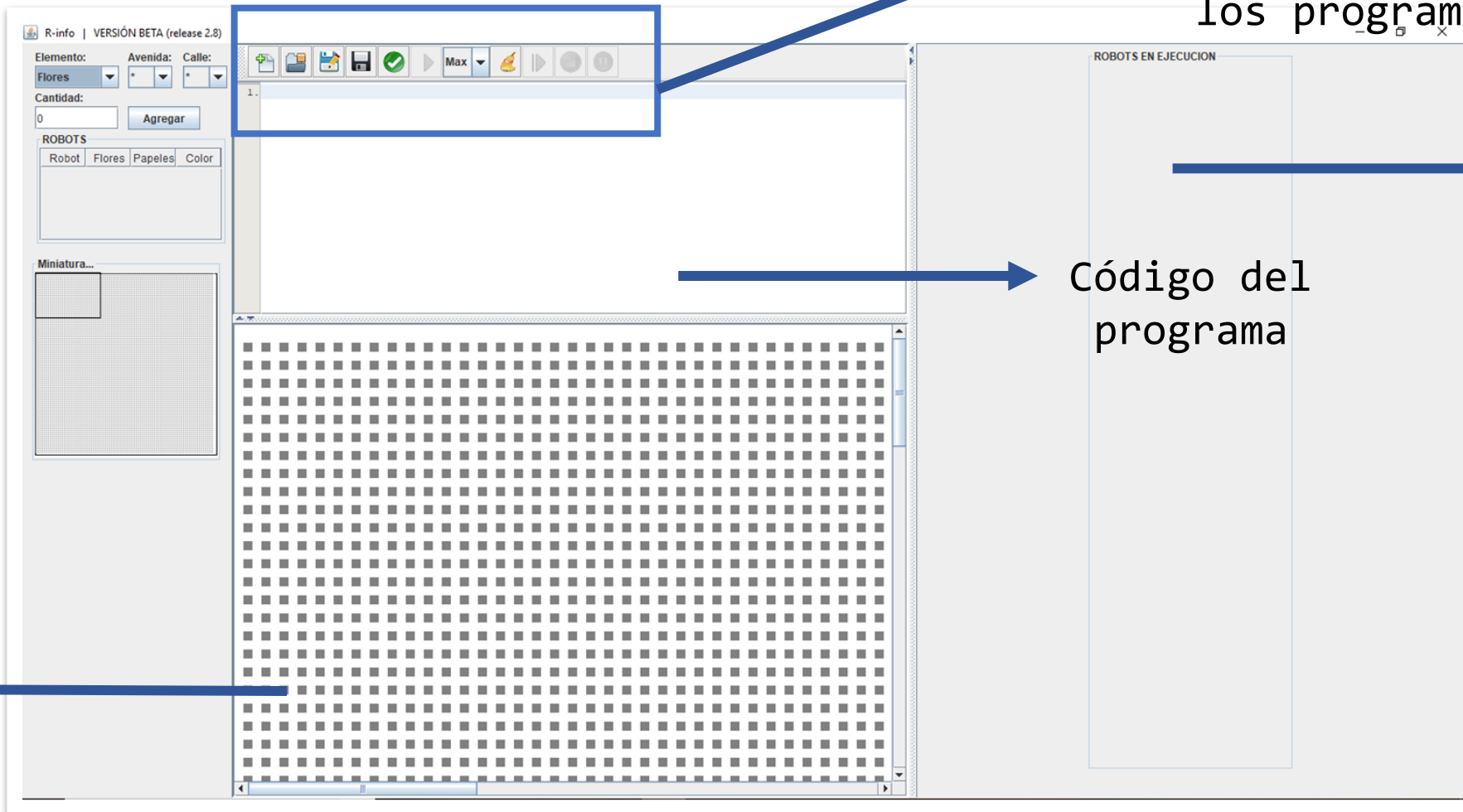
target="_top">
<style type="text/css" media="s">
<link rel="stylesheet" type="text/css" href="css/print.css" media="print">
<link rel="shortcut icon" type="image/x-icon" href="favicon.ico">
<link rel="search" type="application/x-slash" href="http://www.slashdot.org/search.xml">
</script>

```





Ambiente CMRE



Guardar, compilar, ejecutar los programas

Información sobre los robots

Código del programa

Ciudad



Ambiente CMRE

Cómo se relacionan los conceptos de conurrencia con CMRE?



Conceptos

- Recursos Compartidos
- Sincronización
- Procesadores heterogéneos



Ambiente CMRE

Robots

- Se permite declarar más de un robot.

Areas

- Areas privadas, compartidas y parcialmente compartidas

Comunicación y Sincronización

- Enviar y recibir mensajes
- Bloquear y desbloquear esquina



Ambiente CMRE

```
programa nombre

procesos
    // Procesos utilizados por los robots

areas
    // Áreas de la ciudad

robots
    // Robots del programa

variables
    // Variables robots

comenzar
    // Asignación de áreas
    // Inicialización de robots
fin
```

Estructura de un programa



Ambiente CMRE

```
programa nombre
```

```
procesos
```

```
// Procesos utilizados por los robots
```

```
areas
```

```
// Áreas de la ciudad
```

```
robots
```

```
// Robots del programa
```

```
variables
```


```
// Variables robots
```

```
comenzar
```

```
// Asignación de áreas
```

```
// Inicialización de robots
```

```
fin
```



```
proceso nombre (ES flores:numero; E valor:boolean)
```

```
variables
```

```
nombre : tipo
```

```
comenzar
```

```
//código del proceso
```

```
fin
```



Ambiente CMRE

```
programa nombre
```

```
procesos
```

```
// Procesos utilizado
```

```
areas
```

```
// Áreas de la ciudad
```

```
robots
```

```
// Robots del programa
```

```
variables
```

```
// Variables robots
```

```
comenzar
```

```
// Asignación de áreas
```

```
// Inicialización de robots
```

```
fin
```

```
ciudad1: areaC(1,1,10,10) //área Compartida  
ciudad2: areaP(15,15,20,20) //área Privada  
ciudad3: areaPC(30,32,50,51) //área Parcialmente  
compartida
```

areaC
Compartida

Cualquier robot pueden circular por la misma

areaP
Privada

Sólo puede haber en ella un único robot

areaPC
Parc. Comp.

Se debe seleccionar que subconjunto de robots pueden circular por la misma

Ambiente CMRE

R-info | VERSIÓN BETA (release 2.9)

Elemento: Flores Avenida: * Calle: *

Cantidad: 0

Robot	Flores	Papeles	Color
robot1	0	0	Red
robot2	0	0	Blue
robot3	0	0	Magenta

ROBOTS

Miniatura...

```
1. programa areasEjemplo
2. areas
3.   area1: AreaC(1,1,10,10)
4.   area2: AreaP(12,1,15,10)
5.   area3: AreaPC(17,1,30,10)
6. robots
7.   robot florero
8.   variables
9.     avenida:numero
10.    calle:numero
11. comenzar
12.   avenida2:=PosAv
13.   calle2:=PosCa
14. fin
15. variables
16. robot1:florero
```

Área compartida

Área privada

Área parcialmente compartida

ROBOTS EN EJECUCION

robot1 Pos: (00,00)
Bolsa Esquina
F P F P
00 00 00 00
Nuevo

robot2 Pos: (00,00)
Bolsa Esquina
F P F P
00 00 00 00
Nuevo

robot3 Pos: (00,00)
Bolsa Esquina
F P F P
00 00 00 00
Nuevo

Windows taskbar: Escribe aquí para buscar, 8:31, 17/10/2019



Ambiente CMRE

```
programa nombre
```

```
procesos
```

```
// Procesos utilizados por los robots
```

```
areas
```

```
// Áreas de la ciudad
```

```
robots
```

```
// Robots del programa
```

```
variables
```

```
// Variables robots
```

```
comenzar
```

```
// Asignación de áreas
```

```
// Inicialización de robots
```

```
fin
```

```
robot tipo1
```

```
variables
```

```
...
```

```
comenzar
```

```
// Código del robot 1
```

```
fin
```



Ambiente CMRE

```
programa nombre


procesos
    // Procesos utilizados por los robots

areas
    // Áreas de la ciudad

robots
    // Robots del programa

variables
    // Variables robots

comenzar
    // Asignación de áreas
    // Inicialización de robots
fin
```



```
r1: tipo1
r2: tipo1
```



Ambiente CMRE

programa nombre

procesos

// Procesos utilizados por los robots

areas

// Áreas de la ciudad

robots

// Robots del programa

variables


// Variables robots

comenzar

// Asignación de áreas

// Inicialización de robots

fin



r1: tipo1
r2: tipo2



Ambiente CMRE

R-info | VERSIÓN BETA (release 2.9)

Elemento: Flores Avenida: * Calle: *

Cantidad: 0

ROBOTS

Robot	Flores	Papeles	Color
robot1	0	0	
robot2	0	0	
robot3	0	0	

Miniatura...

```
1. programa areasEjemplo
2. areas
3.   area1: AreaC(1,1,10,10)
4.   area2: AreaP(12,1,15,10)
5.   area3: AreaPC(17,1,30,10)
6. robots
7.   robot florero
8.   variables
9.     avenida:numero
10.    calle:numero
11.  comenzar
12.    avenida2:=PosAv
13.    calle2:=PosCa
14.  fin
15. variables
16.   robot1:florero
17.   robot2:florero
18.   robot3:florero
19. comenzar
20.   AsignarArea(robot1,areal)
21.   AsignarArea(robot2,area2)
22.   AsignarArea(robot3,area3)
```

ROBOTS EN EJECUCION

robot1 Pos: (00,00)
Bolsa Esquina
F P F P
00 00 00 00
Nuevo

robot2 Pos: (00,00)
Bolsa Esquina

ESP 8:31
INTL 17/10/2019



Ambiente CMRE

```
programa nombre

procesos
    // Procesos utilizados por los robots


areas
    // Áreas de la ciudad

robots
    // Robots del programa

variables
    // Variables robots

comenzar
    // Asignación de áreas
    // Inicialización de robots
fin
```

Un robot puede estar asignado a 1 o más de un áreas del programa



```
//AsignarArea(variableRobot,nombreArea)
AsignarArea(r1,ciudad1)
iniciar(r1, 5, 5)
```



Actividades en máquina



Para poder realizar esta actividad en el horario de teoría el alumno tiene que haber copiado en su máquina el r-info2.9.

Analice la solución presentada en el **Ejercicio1-a**. Qué hace? Es correcta?.

Analice la solución presentada en el **Ejercicio1-b**. Qué hace? Es correcta?



Analice la solución presentada en el **Ejercicio1-c**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-d**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-e**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-f**. Qué hace? Es correcta?

Analice la solución presentada en el **Ejercicio1-g**. Qué hace? Es correcta?



Actividades en máquina



Realizar los siguientes ejercicios. Los ejercicios deben realizarse de a uno y en el orden que se presentan.



Ejercicio 1-ha: Realice un programa donde un robot recorra el perímetro de un rectángulo de un tamaño 5 (alto) x 3 (ancho) juntando flores. Al finalizar informe las flores juntadas. Inicialmente el robot se encuentra en la esquina (2,2). **Debe modularizar el rectángulo.**



Ejercicio 1-hb: Realice un programa donde dos robots recorren el perímetro de un rectángulo de un tamaño 5 (alto) x 3 (ancho) juntando flores. Al finalizar informe las flores juntadas por cada uno. Inicialmente los robots se encuentran en la esquina (2,2) y (6,2) respectivamente. **Debe modularizar el rectángulo.**



Ejercicio 1-hc: Qué tiene que cambiar en su código si el robot 1 debe realizar un rectángulo de 5 (alto) x 3 (ancho) juntando flores y el robot 2 un rectángulo de 8 (alto) x 2 (ancho) juntando flores.